LINUX
PLUMBERS
CONFERENCE
August 24-28, 2020

Kernel Configurations for Safety

Elana Copperman, PhD –Mobileye / Intel

LINUX
PLUMBERS
CONFERENCE
August 24-28, 2020

Agenda

- Use cases for Linux in safety critical applications
- Criteria for safe kernel configurations
- Security vs safety in kernel configuration
- Next steps

Disclaimer + call for action

LINUX
PLUMBERS
CONFERENCE
August 24-28, 2020

Whoami

- System Safety Architect at Mobileye (part of Intel).
- Supports design of safety features in Mobileye products, including system boot; drivers; and Linux infrastructure.
- Before working at Mobileye, worked as a Security Architect for Cisco-Il (formerly NDS) and more recently as a security consultant for major European automotive concerns on behalf of various Israeli startups.
- Research interests focus on software engineering methodologies and security engineering.

LINUX
PLUMBERS
CONFERENCE
August 24-28, 2020

Safety-critical applications

- Use cases in automotive domain
- Open source and culture shock
- Safety qualification
- ELISA – Embedded Linux In Safety Applications, “bridging the gap”

LINUX
PLUMBERS
CONFERENCE
August 24-28, 2020

Dependability

- Security vs safety: what's in common, what's different
- Kernel self-protection

This is a brain-dump of the various options for a particularly paranoid system.

- Kernel documentation, kernel self-protection

The goals for successful self-protection systems would be that they are effective, on by default, require no opt-in by developers, have no performance impact, do not impede kernel debugging, and have tests. It is uncommon that all these goals can be met, but it is worth explicitly mentioning them, since these aspects need to be explored, dealt with, and/or accepted.

LINUX
PLUMBERS
CONFERENCE
August 24-28, 2020

Security Criteria

- Effective – achieve well defined goals
- Performance – don’t shoot yourself in the foot
- Do not impede kernel debugging
- *On by default*
- *Require no opt-in by developers*
- *Have tests*

Safety Criteria

- Safety is all about **risk management of liability**
- Traceability is fundamental, what cannot be measured does not exist
- Safety will also require well defined and measurable criteria
- Information leakage (security) vs traceability (safety)
- Traceability → log, journal, audit, debug features

LINUX
PLUMBERS
CONFERENCE
August 24-28, 2020

Safety == Security

- Enable CONFIG_BUG_ON_DATA_CORRUPTION
- Disable CONFIG_DEVKMEM
- Enable CONFIG_FORTIFY_SOURCE
- Disable CONFIG_PROC_KCORE

LINUX PLUMBERS CONFERENCE
August 24-28, 2020

Safety != Security

- Enable CONFIG_DEVMEM
- Enable CONFIG_ELF_CORE
- Enable CONFIG_FTRACE_SYSCALLS
- Enable CONFIG_HIBERNATION
- Enable CONFIG_KEXEC
- Enable CONFIG_PROC_PAGE_MONITOR
- Enable CONFIG_STACK_TRACER

LINUX PLUMBERS CONFERENCE
August 24-28, 2020

Classification

- Methodology - [Kernel documentation]
- Attack surface reduction
  - CONFIG_STRICT_KERNEL_RWX, CONFIG_STRICT_MODULE_RWX *Executable code/RO data must not be writable, all data must not be executable*
  - X86 SMEP/SMAP, ARM PXN/PAN *Kernel must never execute user-space memory*
  - seccomp *How to define "unsafe" system calls? How to define "trusted" processes with access to privileged syscalls (e.g., BPF creation or user namespaces)?*

LINUX PLUMBERS CONFERENCE
August 24-28, 2020

Classification (con't)

- Memory integrity
  - Stack buffer overflow - CONFIG_STACKPROTECTOR, CONFIG_STACKPROTECTOR_STRONG,
  - Heap memory integrity – CONFIG_HARDENED_USERCOPY
  - Enable CONFIG_SCHED_STACK_END_CHECK
  - Enable CONFIG_STACKTRACE
  - Define CONFIG_FRAME_WARN size

LINUX PLUMBERS CONFERENCE
August 24-28, 2020

Kernel Memory

- CONFIG_STRICT_KERNEL_RWX
- CONFIG_STRICT_MODULE_RWX
- Module safety
  - Tristate, M configuration
  - Enable CONFIG_MODULES
  - Disable CONFIG_MODULE_FORCE_LOAD
  - Enable CONFIG_MODVERSIONS
  - CONFIG_MODULE_SIG

LINUX
PLUMBERS
CONFERENCE
August 24-28, 2020

Classification (some more)

- Probabilistic defenses
  - KASLR – Enable CONFIG_RANDOMIZE_BASE
  - Heap randomization - Disable CONFIG_COMPAT_BRK
  - Enable CONFIG_SLAB_FREELIST_RANDOM
  - Enable CONFIG_SLAB_FREELIST_HARDENED
- Preventing information exposures
  - Enable CONFIG_SLUB_DEBUG
  - CONFIG_PAGE_POISINING

LINUX PLUMBERS CONFERENCE
August 24-28, 2020

ISO26262 - FFI

- Freedom From Interference

Absence of cascading failures between two or more elements that could lead to the violation of a safety requirement

- Chapter 6, Annex D – Achievement of FFI

➢ **Timing and execution** – blocking of execution, deadlocks, livelocks, incorrect allocation of execution time, incorrect synchronization between software elements

➢ **Memory** – corruption of content, inconsistent data, stack overflow or underflow, read or write access to memory allocated to another software element

➢ **Exchange of information** – repetition/loss/delay/insertion/incorrect addressing/ corruption of information, asymmetric information sent from a sender to multiple receivers, information from a sender received by only a subset of receivers, blocking access to a communication channel

LINUX PLUMBERS CONFERENCE
August 24-28, 2020

Ongoing work

- ISO26262 is domain specific, over-restrictive
- CWE (Common Weakness Enumeration) software development categories from Security → Safety
- Kernel configurations are only a part of the picture
- Focus on test frameworks for safety validation
- Dependability Micro-conference

LINUX PLUMBERS CONFERENCE
August 24-28, 2020

Debug features

- Enable CONFIG_DEBUG_KMEMLEAK
- Enable CONFIG_DEBUG_MEMORY_INIT
- Enable CONFIG_DEBUG_OBJECTS
- Enable CONFIG_DEBUG_PAGEALLOC
- Enable CONFIG_DEBUG_PER_CPU_MAPS
- Enable CONFIG_DEBUG_RODATA
- Enable CONFIG_DEBUG_SECTION_MISMATCH
- Enable CONFIG_DEBUG_STACKOVERFLOW
- Enable CONFIG_DEBUG_VM

LINUX PLUMBERS CONFERENCE
August 24-28, 2020

Passive safety features

- Enable CONFIG_KALLSYMS
- Enable CONFIG_KPROBES
- Enable CONFIG_PAGE_OWNER
- Enable CONFIG_REFCOUNT_FULL
- Enable CONFIG_SLUB_STATS
- Enable CONFIG_STACK_USAGE

LINUX PLUMBERS CONFERENCE
August 24-28, 2020

Lockup and hangs

- Enable CONFIG_DETECT_HUNG_TASK
- Enable CONFIG_SOFTLOCKUP_DETECTOR
- Enable CONFIG_WQ_WATCHDOG

LINUX PLUMBERS CONFERENCE
August 24-28, 2020

Lock mechanisms self-checks

- Enable CONFIG_DEBUG_ATOMIC_SLEEP
- Enable CONFIG_DEBUG_LOCK_ALLOC
- Enable CONFIG_DEBUG_LOCKING_API_SELFTESTS
- Enable CONFIG_DEBUG_MUTEXES
- Enable CONFIG_DEBUG_RT_MUTEXES
- Enable CONFIG_DEBUG_SPINLOCK

LINUX PLUMBERS CONFERENCE
August 24-28, 2020

Lock mechanisms self-checks (con’t)

- Enable CONFIG_LOCK_STAT
- Enable CONFIG_LOCK_TORTURE_TEST
- Enable CONFIG_PROVE_LOCKING
- Enable CONFIG_RCU_TORTURE_TEST
- Enable CONFIG_WW_MUTEX_SELFTEST
- Enable CONFIG_DEBUG_WW_MUTEX_SLOWPATH

Vision: Safety Policy

- **Security policies** are a set of rules which are used to configure security decisions such as access control in a particular context.
- Similarly, we may define **safety policies** as a set of rules which may be used to configure safety mechanisms in a particular use case.
- The safety policy may define features such as ASIL level and specific constraints on communication between hardware and/or software elements.

LINUX
PLUMBERS
CONFERENCE
August 24-28, 2020

Safety Policy

- Based on the designated safety policy, the kernel image can be built with a pre-defined set of default configurations.
- The chosen default configurations will be based on clear criteria and will be pre-assessed for safety qualification
- Any deviation from default policy in a specific use case will require justification

LINUX PLUMBERS CONFERENCE
August 24-28, 2020

LSaMs = Linux Safety Modules

- This methodology can be extended.
- Long-term, we can aim to define “Linux Safety Modules” as building blocks for safety-critical applications.
- LSaMs are pre-qualified out of context but provide tools, criteria and guidelines for design and integration in the specific context of the use case.
- The safety qualification for the final product is expected to be based on those building blocks.

LINUX
PLUMBERS
CONFERENCE
August 24-28, 2020

Next steps

- Safety Policies
- Classifications of existing kernel configurations
- New features / configurations relevant to specific safety policies
- Long-term deployment and maintenance

elana.copperman@mobileye.com